

**CLAIMS:**

What is claimed is:

- Sub 1  
Aa 2
1. A method of verifying the integrity of unauthenticated code, comprising:
    - 3 receiving automatically authenticated code, the
    - 4 automatically authenticated code including an embedded first hash
    - 5 value of the unauthenticated code;
    - 6 receiving the unauthenticated code;
    - 7 generating a second hash value of the unauthenticated
    - 8 code;
    - 9 comparing the first hash value and the second hash
    - 10 value; and
    - 11 verifying the integrity of the unauthenticated code if
    - 12 the first hash value and the second hash value match.
  2. The method of claim 1, wherein the automatically authenticated code is compiled Java code and wherein the unauthenticated code is native code.
  3. The method of claim 1, wherein the automatically authenticated code is a Java application or applet and wherein the unauthenticated code is a dynamically linked library.
  4. The method of claim 1, wherein the first hash value is obtained using a hashing function and wherein generating a second hash value of the unauthenticated code includes using the same hashing function as was used to obtain the first hash value.

1 5. The method of claim 4, wherein the hashing function is  
2 identified based on information stored in the automatically  
3 authenticated code.

1 6. The method of claim 1, further comprising:  
2 executing the automatically authenticated code using a  
3 virtual machine; and  
4 sending a request to a server from which the  
5 automatically authenticated code was received, the request being  
6 for the unauthenticated code.

1 7. The method of claim 1, wherein, if the first hash value  
2 and the second hash value do not match, the method further  
3 comprises:  
4 receiving the unauthenticated code again;  
5 generating a third hash value of the unauthenticated  
6 code; and  
7 comparing the first hash value and the third hash  
8 value.

1 8. The method of claim 7, wherein if the third hash value  
2 and the first hash value do not match, the method further  
3 comprises:  
4 comparing the second hash value and the third hash  
5 value; and  
6 if the second hash value and the third hash value

7 match, determining that the unauthenticated code has been  
8 corrupted intentionally.

1 9. The method of claim 8, wherein if the second hash value  
2 and the third hash value do not match, it is determined that the  
3 unauthenticated code has been corrupted unintentionally.

1 10. The method of claim 1, wherein the method is  
2 implemented in a virtual machine associated with a web browser on  
3 a client device.

1 11. An apparatus for verifying the integrity of  
2 unauthenticated code, comprising:

3 a virtual machine; and

4 an unauthenticated code verification element, wherein  
5 the virtual machine receives automatically authenticated code,  
6 the automatically authenticated code including an embedded first  
7 hash value of the unauthenticated code, and receives the  
8 unauthenticated code, and wherein

9 the unauthenticated code verification element generates  
10 a second hash value of the unauthenticated code, compares the  
11 first hash value and the second hash value, and verifies the  
12 integrity of the unauthenticated code if the first hash value and  
13 the second hash value match.

1 12. The apparatus of claim 11, wherein the automatically  
2 authenticated code is compiled Java code and wherein the

3 {unauthenticated code is native code.

1 13. The apparatus of claim 11, wherein the automatically  
2 authenticated code is a Java application or applet and wherein  
3 the unauthenticated code is a dynamically linked library.

1 14. The apparatus of claim 11, wherein the first hash value  
2 is obtained using a hashing function and wherein the  
3 unauthenticated code verification element generates a second hash  
4 value of the unauthenticated code using the same hashing function  
5 as was used to obtain the first hash value.

1 15. The apparatus of claim 14, wherein the hashing function  
2 is identified by the unauthenticated code verification element  
3 based on information stored in the automatically authenticated  
4 code.

1 16. The apparatus of claim 11, wherein the virtual machine  
2 executes the automatically authenticated code and  
3 sends a request to a server from which the automatically  
4 authenticated code was received, the request being for the  
5 unauthenticated code.

1 17. The apparatus of claim 11, wherein, if the first hash  
2 value and the second hash value do not match, the virtual machine  
3 receives the unauthenticated code again, the unauthenticated code  
4 verification element generates a third hash value of the

5 unauthenticated code and compares the first hash value and the  
6 third hash value.

1 18. The apparatus of claim 17, wherein if the third hash  
2 value and the first hash value do not match, the unauthenticated  
3 code verification element compares the second hash value and the  
4 third hash value and, if the second hash value and the third hash  
5 value match, determines that the unauthenticated code has been  
6 corrupted intentionally.

7 19. The apparatus of claim 18, wherein if the second hash  
8 value and the third hash value do not match, the unauthenticated  
9 code verification element determines that the unauthenticated  
0 code has been corrupted unintentionally.

1 20. The apparatus of claim 11, wherein the virtual machine  
2 and the unauthenticated code verification element are associated  
3 with a web browser on a client device.

4 21. A computer program product in a computer readable  
5 medium for verifying the integrity of unauthenticated code,  
6 comprising:

7 first instructions for receiving automatically  
8 authenticated code, the automatically authenticated code  
9 including an embedded first hash value of the unauthenticated  
0 code;

1 second instructions for receiving the unauthenticated

9 code;  
10 third instructions for generating a second hash value  
11 of the unauthenticated code;  
12 fourth instructions for comparing the first hash value  
13 and the second hash value; and  
14 fifth instructions for verifying the integrity of the  
15 unauthenticated code if the first hash value and the second hash  
16 value match.

22. The computer program product of claim 21, wherein the  
automatically authenticated code is compiled Java code and  
wherein the unauthenticated code is native code.

23. The computer program product of claim 21, wherein the  
automatically authenticated code is a Java application or applet  
and wherein the unauthenticated code is a dynamically linked  
library.

24. The computer program product of claim 21, wherein the  
first hash value is obtained using a hashing function and wherein  
the third instructions for generating a second hash value of the  
unauthenticated code include instructions for using the same  
hashing function as was used to obtain the first hash value.

25. The computer program product of claim 24, further  
comprising instructions for identifying the hashing function  
based on information stored in the automatically authenticated

4 code.

1 26. The computer program product of claim 21, further  
2 comprising:

3 sixth instructions for executing the automatically  
4 authenticated code using a virtual machine; and

5 seventh instructions for sending a request to a server  
6 from which the automatically authenticated code was received, the  
7 request being for the unauthenticated code.

8 27. The computer program product of claim 21, further  
9 comprising:

10 sixth instructions for receiving the unauthenticated  
11 code again, if the first hash value and the second hash value do  
12 not match;

13 seventh instructions for generating a third hash value  
14 of the unauthenticated code; and

15 eighth instructions for comparing the first hash value  
16 and the third hash value.

17 28. The computer program product of claim 27, further  
18 comprising:

19 ninth instructions for comparing the second hash value  
20 and the third hash value, if the third hash value and the first  
21 hash value do not match; and

22 tenth instructions for determining that the  
23 unauthenticated code has been corrupted intentionally, if the

8 second hash value and the third hash value match.

1 29. The computer program product of claim 28, further  
2 comprising eleventh instructions for determining that the  
3 unauthenticated code has been corrupted unintentionally, if the  
4 second hash value and the third hash value do not match.

005150 061500 005150 061500